

WEB- ARDUINO

Рассматривается возможность построения web-сервера для управления оборудованием и получением данных с различных датчиков через сеть Интернет. Указывается на то, что для удаленного управления необходимо проводить авторизацию. В противном случае оборудование будет не защищено от несанкционированного доступа пользователей сети Интернет. Отмечается, что управляющие web-сервера создаются на базе микроконтроллеров, которые имеют малые ресурсы и не в состоянии работать с протоколами HTTPS, SSL, TLS. Поэтому эти сервера являются уязвимыми со стороны сетевых атак. В работе рассмотрено создание web-сервера на Arduino, который использует авторизацию на основе GET и POST запросов, а также модернизированную HTTP basic authentication. Модернизация состоит в том, что для авторизации используется пароль из списка паролей, который выбирается пользователем на основании ключа, пересылаемого сервером. При каждом новом входе на сервер предыдущий пароль становится недействительным. Представлен практический пример web-сервера на Arduino Mega, на котором установлены три светодиода, имитирующие включение-выключение 3-х силовых источников питания (например, электророзеток), датчик температуры DS18B20, датчик влажности и температуры DHT 11 и барометрический датчик BMP280. Сервер тестировался с тремя контроллерами Ethernet: enc28j60, W5100 и w5500. Для этого использовались три библиотеки: UIPEthernet, Ethernet и Ethernet2, которые показали одинаковые результаты работы. Установлено, что авторизация на сервере Arduino с использованием метода запроса GET является нецелесообразной, так как при использовании форм ввода с полями для пароля коды открыто высвечиваются в адресной строке. Авторизация с использованием метода POST скрывает передаваемые коды. Но коды передаются по сети в открытой форме, поэтому их можно перехватить с помощью программ sniffеров. Использование протоколов HTTPS, SSL, TLS позволяет сделать такую авторизацию безопасной. Программа, реализующая web-сервер Arduino, составлена в среде разработки Arduino IDE ver. 1.8.6.

Ключевые слова: Arduino, Ethernet Shield Arduino, ENC28J60, W5100, W5500, протокол HTTPS, SSL, TLS, GET и POST запросы, base64-encoded, basic authentication, web-server, микроконтроллер.

A.A. MYASISHEV
Khmelnitsky National University

AUTHORIZATION ON WEB SERVER ARDUINO FOR CONTROLLING THROUGH THE INTERNET

The possibility of building a web server for controlling equipment and obtaining data from various sensors via the Internet is considered. It is indicated that authorization is necessary for remote management. Otherwise, the equipment will not be protected from unauthorized access of Internet users. It is noted that the managing web servers are created on the basis of microcontrollers, which have small resources and are not able to work with the protocols HTTPS, SSL, TLS. Therefore, these servers are vulnerable to network attacks. The paper considers the creation of a web server on Arduino, which uses authorization based on GET and POST requests, as well as modernized HTTP basic authentication. The upgrade consists in the fact that the password is used for authorization from the password list, which is selected by the user based on the key sent by the server. With each new login to the server, the previous password becomes invalid. A practical example of a web server is presented on the Arduino Mega, on which are installed three LEDs simulating switching on / off of 3 power supply sources (for example, power outlets), DS18B20 temperature sensor, humidity and temperature sensor DHT 11 and barometric sensor BMP280. The server was tested with three Ethernet controllers: enc28j60, W5100 and w5500. For this purpose, three libraries were used: UIPEthernet, Ethernet and Ethernet2, which showed the same results. It has been established that authorization on the Arduino server using the GET request method is inappropriate, because when using input forms with password fields, the codes are displayed in the address bar. Authorization using the POST method hides the transmitted codes. But the codes are transmitted over the network in an open form, so they can be intercepted using sniffers programs. Using HTTPS, SSL, TLS allows you to make such authorization secure. The program that implements the Arduino web server is made in the Arduino IDE 1.8.6.

Keywords: Arduino, Ethernet Shield Arduino, ENC28J60, W5100, W5500, HTTPS, SSL, TLS, GET and POST requests, base64-encoded, basic authentication, web-server, microcontroller.

web-

Arduino.
(shield),

Arduino

Arduino

web- Arduino

« » SSL TLS,

Arduino Ethernet Shield Arduino

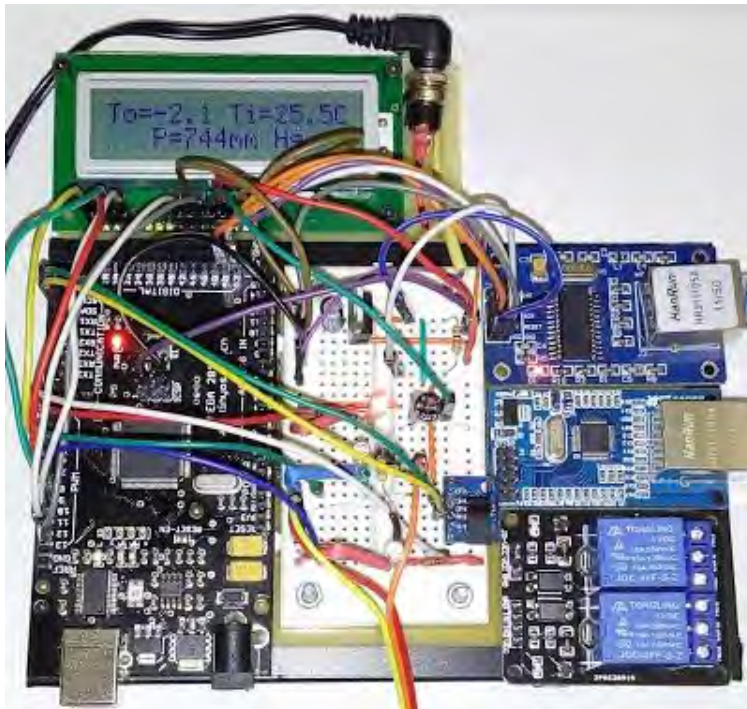
TCP/IP. GET POST

HTTP authentication, Arduino. HTTP 1.0/1.1.

web- Arduino, (

DHT11(BMP280). 1 Arduino Mega

ENC28J60 (W5500), LCD



.1. web- Arduino Mega

GET POST

- 11361
- 11360
- 11381
- 11380
- 11401
- 11400

GET

GET / HTTP/1.1

Accept: text/html, application/xhtml+xml, image/jxr, */*

Accept-Language: ru,en-US;q=0.8,en;q=0.5,uk;q=0.3

User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; Trident/7.0; MALNJS; rv:11.0) like Gecko

Accept-Encoding: gzip, deflate

Host: 192.168.1.18:81

Connection: Keep-Alive

\r\n.

[1]:

GET /?ron=11361&roff=&gon=&goff=&bon=11401&boff= HTTP/1.1

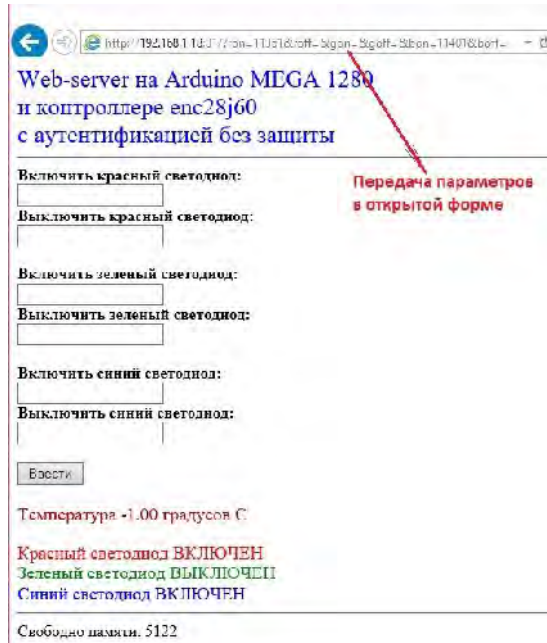
Accept: text/html, application/xhtml+xml, image/jxr, */*

Referer: http://192.168.1.18:81/

Accept-Language: ru,en-US;q=0.8,en;q=0.5,uk;q=0.3
 User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; Trident/7.0; MALNJS; rv:11.0) like Gecko
 Accept-Encoding: gzip, deflate
 Host: 192.168.1.18:81
 Connection: Keep-Alive

GET
 2

[1].



. 2.

GET

POST

POST

POST / HTTP/1.1

Accept: text/html, application/xhtml+xml, image/jxr, */*
 Referer: http://192.168.1.18:81/
 Accept-Language: ru,en-US;q=0.8,en;q=0.5,uk;q=0.3
 Content-Type: application/x-www-form-urlencoded
 User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; Trident/7.0; MALNJS; rv:11.0) like Gecko
 Accept-Encoding: gzip, deflate
 Host: 192.168.1.18:81
 Content-Length: 42
 Connection: Keep-Alive
 Cache-Control: no-cache

<--- --->
 ron=11361&roff=&gon=&goff=&bon=11401&boff=

. 3

POST

POST

[2].

```
while(client.available()) // POST
{ post = client.read();
  if (buffer.length() <= bufferMax) { buffer += post; } }
```

buffer

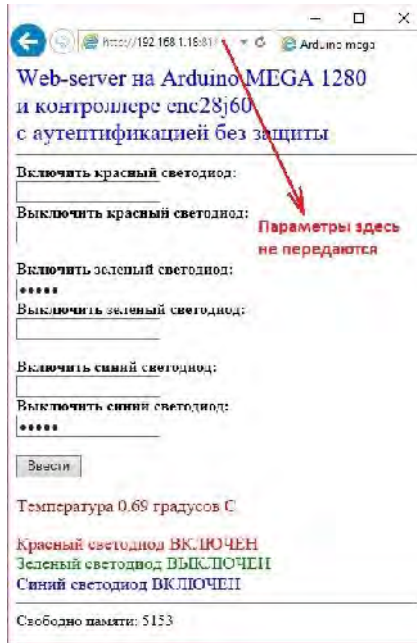
(ron=11361&roff=&gon=&goff=&bon=11401&boff=).

```
if(buffer.indexOf("ron=11361") >= 0) {digitalWrite(3, HIGH);}
if(buffer.indexOf("roff=11360") >= 0) {digitalWrite(3, LOW);}
if(buffer.indexOf("gon=11381") >= 0) {digitalWrite(5, HIGH);}
if(buffer.indexOf("goff=11380") >= 0) {digitalWrite(5, LOW);}
```

```

if(buffer.indexOf("bon=11401") >= 0) {digitalWrite(7, HIGH);}
if(buffer.indexOf("boff=11400") >= 0) {digitalWrite(7, LOW);}
(
).

```



. 3. POST

(, POST)

sniffer

https (ssl, tsl)

HTTP authentication.

HTTP 1.0/1.1,

web-

HTTP

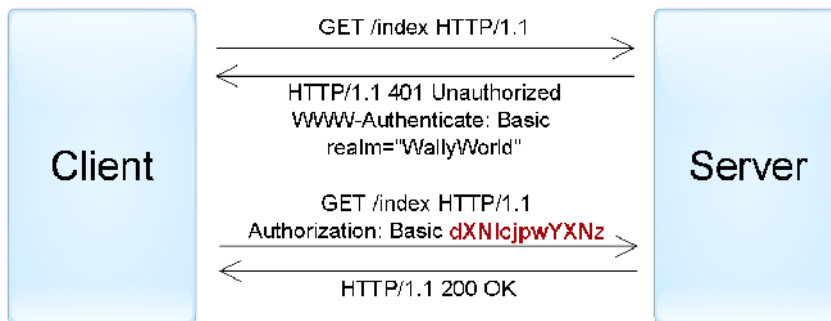
1. "401 Unauthorized" "WWW-Authenticate"

2. , username

password.

3. web-

1. Basic – username password Authorization
 (base64-encoded). HTTPS (HTTP over
 SSL) , basic (. 4).



. 4. Basic

2. Digest — challenge-response- , nonce,
 MD5 , nonce.

attacks (Basic basic). , man-in-the-middle

3. NTLM (Windows authentication) — challenge-response HTTP, Windows Active Directory pass-the-hash basic,

Arduino. 4 Arduino :
GET / HTTP/1.1
Host: 192.168.1.18:81
Connection: keep-alive
Cache-Control: max-age=0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,/*;q=0.8*
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/44.4.2403.3 Amigo/44.4.2403.3 MRCHROME SOC Safari/537.36
Accept-Encoding: gzip, deflate, sdch
Accept-Language: ru-RU,ru;q=0.8,en-US;q=0.6,en;q=0.4

HTTP/1.0 401 Unauthorized
WWW-Authenticate: Basic realm="Arduino - HOME"
 Amigo :
GET / HTTP/1.1
Host: 192.168.1.18:81
Connection: keep-alive
Cache-Control: max-age=0
Authorization: Basic YWxleDoxMzY=
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,/*;q=0.8*
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/44.4.2403.3 Amigo/44.4.2403.3 MRCHROME SOC Safari/537.36
Accept-Encoding: gzip, deflate, sdch
Accept-Language: ru-RU,ru;q=0.8,en-US;q=0.6,en;q=0.4

Arduino html
HTTP/1.0 200 OK
Content-Type: text/html
 < html >
 YWxleDoxMzY= Base64 alex:136 (alex – login, 136 – password, 5)

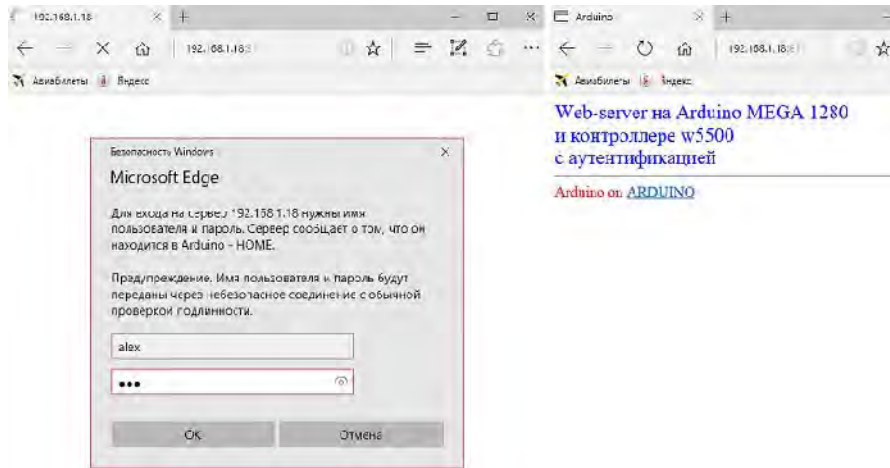
Arduino.
 [3].

```

if (readString.lastIndexOf("YWxleDoxMzY=")>-1) {
  if (readString.lastIndexOf("GET /favicon.ico")>-1) {
    client.println("HTTP/1.0 404 Not Found");
  } else html_doc(client); }
else { client.println("HTTP/1.0 401 Unauthorized");
  client.println("WWW-Authenticate: Basic realm=\"Arduino - HOME\"");}
    
```

 (YWxleDoxMzY=) Base64 decode.

[3] web- (), DS18B20 () DHT 11 (3) : : 1. alex:136; 2. alex:138; 3. alex:140 BMP280.



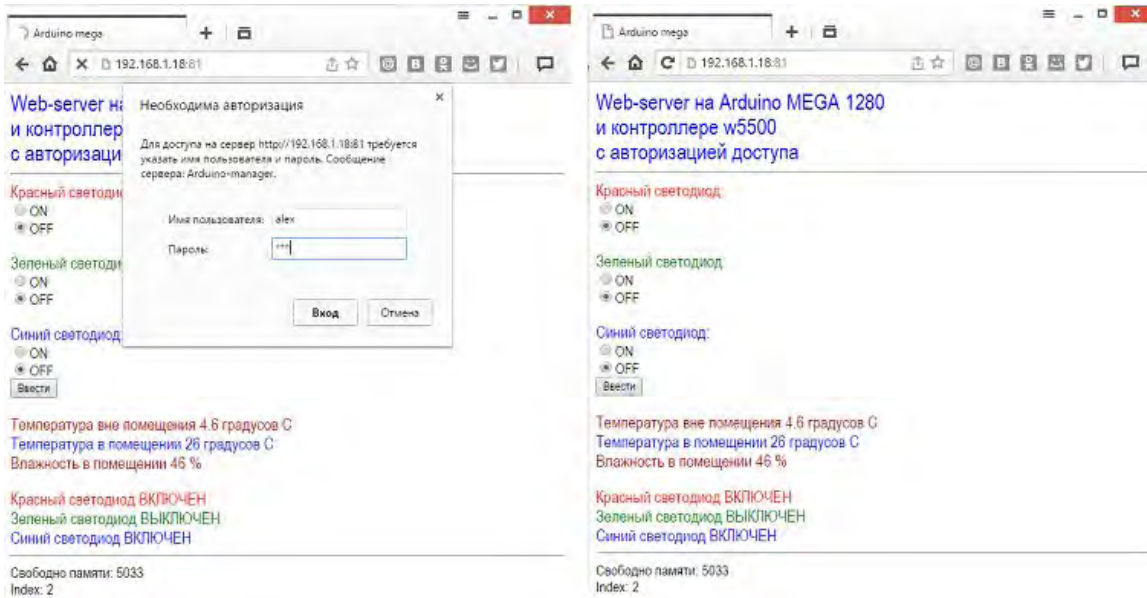
.5.

Arduino

index.
http://192.168.1.18:81/index.

6

Arduino Mega.



.6.

web- Arduino Mega

Index:2
alex:138

```
// 1-
else if (readString.lastIndexOf("YWxleDoxMzY=") > -1 && passwd == 1) {
    passwd = 2; if (readString.lastIndexOf("GET /favicon.ico") > -1) {
        client.println("HTTP/1.0 404 Not Found"); }
    else { onoff(buffer); html_doc(client); } }
// 2-
else if (readString.lastIndexOf("YWxleDoxMzg=") > -1 && passwd == 2) {
    passwd = 3; if (readString.lastIndexOf("GET /favicon.ico") > -1) {
        client.println("HTTP/1.0 404 Not Found"); }
    else { onoff(buffer); html_doc(client); } }
// 3-
...

```

POST
web- Arduino:

POST / HTTP/1.1
Accept: text/html, application/xhtml+xml, image/jxr, */*

Referer: http://192.168.1.18:81/
 Accept-Language: ru,en-US;q=0.8,en;q=0.5,uk;q=0.3
 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/51.0.2704.79 Safari/537.36 Edge/14.14393
 Content-Type: application/x-www-form-urlencoded
 Accept-Encoding: gzip, deflate
 Host: 192.168.1.18:81
 Content-Length: 23
 Connection: Keep-Alive
 Cache-Control: no-cache
 Authorization: Basic YWxleDoxMzY=
 < >
 r=0&g=1&b=0&av=2018year

POST.

```

        (
            r, g b. r=1 -
        (g) (b).
    :
    if(buffer.indexOf("r=1") >= 0) {digitalWrite(3, HIGH);}
    if(buffer.indexOf("r=0") >= 0) {digitalWrite(3, LOW);}
    if(buffer.indexOf("g=1") >= 0) {digitalWrite(5, HIGH);}
    if(buffer.indexOf("g=0") >= 0) {digitalWrite(5, LOW);}
    if(buffer.indexOf("b=1") >= 0) {digitalWrite(7, HIGH);}
    if(buffer.indexOf("b=0") >= 0) {digitalWrite(7, LOW);}
    
```

```

        ( 2018year ),
    client.println("<input type='hidden' name='av' value='2018year'>");
        hidden, PUT
    2018year av (r=0&g=1&b=0&av=2018year).
    else if (buffer.lastIndexOf("2018year")>-1 ) {
        if (readString.lastIndexOf("GET /favicon.ico")>-1) {
            client.println("HTTP/1.0 404 Not Found"); }
        else { onoff(); html_doc(client); } }
    
```

Arduino

1. Arduino GET
2. POST
3. HTTP Basic username password HTTPS (HTTP over SSL), (base64-encoded).
4. Basic
5. Arduino IDE 1.8.6 index. web- Arduino Mega

1. Web-server Arduino MEGA w5500 [] -
2. GET POST web- Arduino [] - : https://sites.google.com/site/webstm32/get-post-arduino, 2018.
3. Web-server Arduino mega POST [] - : https://sites.google.com/site/webstm32/web-server--post-auten,

- 2018.
4. Kitsum. Web []. – :
<https://it4it.club/topic/13-avtorizaciya-na-web-servere-mikrokontrollera/>, 2015.
 5. Grankin S.S. GET i POST autentifikaciya na web-servere Arduino. [Electronic resource]. - Mode of access: <http://cxem.net/arduino/arduino176.php>, 2016.
 6. Vyrostkov D. Obzor sposobov i protokolov autentifikacii v veb-prilozheniyah. / DataArt Tekhnologicheskij konsalting i razrabotka PO. - [Electronic resource]. - Mode of access: <https://habrahabr.ru/company/dataart/blog/262817>, 2015.
 7. Igo T. Arduino, datchiki i seti dlya svyazi ustrojstv: Per. s angl. - 2-e izd. - SPb.: BHV-Peterburg, 2015. - 544 s. il.
 8. Ivanov P.I. Upravlenie blokom rele cherez brauzer s pomoshch'yu Arduino i Ethernet shield. [Electronic resource]. - Mode of access: <http://www.psub.net/Publication/Details/60>, 2015.
 9. Sklyarov D.V. Iskusstvo zashchity i vzloma informacii. - SPb.: BHV-Peterburg, 2004. - 288 s. il.
 10. Anisimov V.V. Kriptograficheskie metody zashchity informacii. [Electronic resource]. - Mode of access: <https://sites.google.com/site/anisimovkhv/learning/kripto/lecture>, 2012.
 11. Malkov A. Klassifikaciya mekhanizmov autentifikacii pol'zovatelej i ih obzor. [Electronic resource]. - Mode of access: <https://habrahabr.ru/post/177551/>, 2013.
 12. Gavrikov V. SHifrovanie dannyh: kriptozashchita STM32. [Electronic resource]. - Mode of access: <https://www.compel.ru/lib/ne/2016/10/2-shifrovanie-dannyih-kriptozashchita-stm32>, 2016.
 13. SHumel' V.V., Rudikova L.V. Obschie principy organizacii http-servera na mikrokontrollere. [Electronic resource]. - Mode of access: <http://www.elib.bsu.by/bitstream/123456789/52523/1/41-45.pdf>, 2013.

References

1. Myasishchev A.A. Web-server na Arduino MEGA i kontrollere seti w5500 s avtorizaciej dostupa dlya udalennogo upravleniya. [Electronic resource]. - Mode of access: <https://sites.google.com/site/webstm32/web-server-avtorizaciej>, 2018.
2. Myasishchev A.A. GET i POST autentifikaciya na web-servere Arduino. [Electronic resource]. - Mode of access: <https://sites.google.com/site/webstm32/get-post-arduino>, 2018.
3. Myasishchev A.A. Web - server na Arduino mega s prostoju autentifikaciej, ispol'zuyushchej POST zapros. [Electronic resource]. - Mode of access: <https://sites.google.com/site/webstm32/web-server--post-auten>, 2018.
4. Kitsum. Avtorizaciya na Web servere mikrokontrollera. [Electronic resource]. - Mode of access: <https://it4it.club/topic/13-avtorizaciya-na-web-servere-mikrokontrollera/>, 2015.
5. Grankin S.S. Podklyuchenie Arduino k Internetu: nastrojka rezhima klient-server, obrabotka GET i POST zaprosov. [Electronic resource]. - Mode of access: <http://cxem.net/arduino/arduino176.php>, 2016.
6. Vyrostkov D. Obzor sposobov i protokolov autentifikacii v veb-prilozheniyah. / DataArt Tekhnologicheskij konsalting i razrabotka PO. - [Electronic resource]. - Mode of access: <https://habrahabr.ru/company/dataart/blog/262817>, 2015.
7. Igo T. Arduino, datchiki i seti dlya svyazi ustrojstv: Per. s angl. - 2-e izd. - SPb.: BHV-Peterburg, 2015. - 544 s. il.
8. Ivanov P.I. Upravlenie blokom rele cherez brauzer s pomoshch'yu Arduino i Ethernet shield. [Electronic resource]. - Mode of access: <http://www.psub.net/Publication/Details/60>, 2015.
9. Sklyarov D.V. Iskusstvo zashchity i vzloma informacii. - SPb.: BHV-Peterburg, 2004. - 288 s. il.
10. Anisimov V.V. Kriptograficheskie metody zashchity informacii. [Electronic resource]. - Mode of access: <https://sites.google.com/site/anisimovkhv/learning/kripto/lecture>, 2012.
11. Malkov A. Klassifikaciya mekhanizmov autentifikacii pol'zovatelej i ih obzor. [Electronic resource]. - Mode of access: <https://habrahabr.ru/post/177551/>, 2013.
12. Gavrikov V. SHifrovanie dannyh: kriptozashchita STM32. [Electronic resource]. - Mode of access: <https://www.compel.ru/lib/ne/2016/10/2-shifrovanie-dannyih-kriptozashchita-stm32>, 2016.
13. SHumel' V.V., Rudikova L.V. Obschie principy organizacii http-servera na mikrokontrollere. [Electronic resource]. - Mode of access: <http://www.elib.bsu.by/bitstream/123456789/52523/1/41-45.pdf>, 2013.

/Peer review : 28.02.2018 .

/Printed :23.03.2018 .